Progress in Artificial Intelligence
https://doi.org/10.1007/s13748-018-0144-x

A GRASP algorithm for *Quota* sequences with minimum work overload and forced interruption of operations in a mixed product assembly line

Joaquín Bautista[1], Rocío Alfaro-Pozo[2]

[1] IOC ETSEIB Universidad Politécnica de Catalunya
Av. Diagonal 647, 08028, Barcelona, Spain
joaquin.bautista@upc.edu

[2] EAE Business School
C/Aragó, 55, 08015, Barcelona, Spain
ralfaro@eae.es

Summary

We present a GRASP algorithm to solve a problem that involves the sequencing of mixed products in an assembly line. The objective of the problem is to obtain a manufacturing sequence of models that generates a minimum work overload with a forced interruption of operations, that is regular in production, and in which the production mix maintains the *Quota* property in the whole sequence. The implemented GRASP is compared with other procedures using instances of a case study of the Nissan engine manufacturing plant in Barcelona.

Keywords: GRASP; Mixed product lines; *Quota* sequences: Work overload.

1 Introduction

MMSP-W ([1], [2], [3]) (*Mixed Model Sequencing Problem with Workload Minimization*) is a problem of sequences in assembly lines [4] that consists of establishing a bijection between the elements of a set T of manufacturing cycles $(t = 1,..,T)$ and those of a set Ψ of products (T elements). The elements of the set Ψ can be grouped into exclusive classes ψ_i that satisfy the following: $\Psi = \bigcup_{i \in I} \psi_i$ and $\psi_i \cap \psi_{i'} = \emptyset, \forall \{i, i'\} \in I$, where I is the set of product types $(i = 1,..,n)$.

The units of Ψ pass, one after the other, through a set of workstations, K, that are arranged in series (a production line). A unit of type $i \in I$ requires a processing time $p_{i,k}$ for normal activity $(\alpha^N = 1)$ at workstation $k \in K$ $(k = 1,..,m)$.

It is evident that the difference between the classes ψ_i (4x4, vans, trucks) makes the times $p_{i,k}$ heterogeneous, while the cycle time c is identical at every workstation $k \in K$; therefore, the operators and robots (processors) have the same time to complete an operation regardless of the product and the station. This discrepancy between the cycle time and the processing times places the processors between two undesirable situations: (1) delays with idle time and (2) blockages due to work overload.

To alleviate the blockages due to work overload, the processors are eventually granted a time to work that is longer than the cycle time, called the temporary window l_k $(l_k > c)$, which depends on the station $k \in K$ ([5], [6]). This concession can reduce the operator's available time to work on the next product and can also reduce the available time for the next station $(k + 1)$ to work on that product when it is released by station k. When the temporary window is not sufficient to complete the work required by the product, it will be incomplete and will pass to the next station, which results in an interruption of the operation that generates an inefficiency that implies that we call work overload or have the production fall.

The interruption of an operation can be *forced* or *free*. The first occurs when the operator reaches the limit of the temporary window l_k without completing the operation ([6], [7]). The free interruption occurs when the incomplete product passes to the next station before reaching the limit of the temporary window of the emitting station ([8], [9], [10]). Obviously, the operation is interrupted whenever the limit established by the temporary window l_k $(\forall k \in K)$ is reached.

Whether the interruption of the operations is forced or free, the ultimate purpose of the MMSP-W is to obtain a sequence of models that minimizes the total work overload in the $W(T)$ line, which is equivalent to maximizing the total completed work $V(T)$ (Theorem 1 in [6]).

Completing the maximum possible work is not the only desirable objective when establishing a product manufacturing sequence. Certainly, in production environments that are governed by the Just-in-Time manufacturing ideals ([11], [12]) or Lean Manufacturing [13], both of which are linked to the Automotive sector, the manufacturing sequences must have properties that are linked to the regularity of production. The incorporation of the regularity concept in manufacturing sequence problems can be characterized in at least two ways:

cr.1 Restrictions: Imposing minimum and maximum manufacturing levels on various mixed models in each production cycle, with the purpose of maintaining the flow without turbulence for both products and components [8].

cr.2 Objective function: Maximizing the constancy of the product manufacturing rates and/or the component consumption rates with the purpose of minimizing the maximum stock levels of the latter [14].

Obviously, there is also the possibility of establishing a mixed characterization of the regularity, which incorporates into the sequence models both an objective function as well as restrictions on the manufacture of products and/or the consumption of their components [15].

In this work, we will add to the genuine MMSP-W problem the two previous characterizations to achieve sequences with minimum work overload and with some properties that propitiate the regularity of production.

The remainder of this document has the following structure. Section 2 is dedicated to establishing some concepts about regular sequences in production. In section 3, we present the problem under study, which we call MMSP-W/Δ_{Q_X} with preservation of the production mix and forced interruption of operations. In section 4, we describe the GRASP algorithm that was designed for this work. In section 5, we present a case study with its data, the procedures used and their results. Finally, in section 6, we offer some conclusions about this work.

2 Quota property in a manufacturing sequence

Similar to [7], [9] and [10], we will indicate that the sequence $\pi_\varepsilon(T) = (\pi_{1,\varepsilon}, \dots, \pi_{T,\varepsilon})$, which is composed of T units of products and associated with the demand plan ε, has the property of preservation of the production mix if the set of restrictions (1) (pmr: Production Mix Restrictions) is satisfied:

$$\lfloor \lambda_{i,\varepsilon} t \rfloor \leq X_{i,t,\varepsilon} \leq \lceil \lambda_{i,\varepsilon} t \rceil \quad \forall i \in I, \forall t \in \mathrm{T}, \forall \varepsilon \in \mathrm{E}, \text{ con } X_{i,T,\varepsilon} = d_{i,\varepsilon} \; \forall i \in I, \forall \varepsilon \in \mathrm{E} \tag{1}$$

Where:

- I: set of product types, $i = 1, .., |I|$
- E: set of demand plans, $\varepsilon = 1, .., |\mathrm{E}|$
- T: set of manufacturing cycles in every demand plan, $t = 1, .., |\mathrm{T}|$; $T \equiv |\mathrm{T}|$
- $d_{i,\varepsilon}$: demand for units of type $i \in I$ in plan $\varepsilon \in \mathrm{E}$
- $\lambda_{i,\varepsilon}$: proportion of units of type $i \in I$ in plan $\varepsilon \in \mathrm{E}$: $\lambda_{i,\varepsilon} = d_{i,\varepsilon}/T \; \forall i \in I, \forall \varepsilon \in \mathrm{E}$
- $X_{i,t,\varepsilon}$: number of units of type $i \in I$ in the partial sequence $\pi_\varepsilon(t) = (\pi_{1,\varepsilon}, \dots, \pi_{t,\varepsilon}) \subseteq \pi_\varepsilon(T)$ of plan $\varepsilon \in \mathrm{E}$: Actual production associated with the partial sequence $\pi_\varepsilon(t)$.

Note that the constraints pmr (1) impose that the actual production $X_{i,t,\varepsilon}$, for every product $i \in \mathrm{I}$, every manufacturing cycle $t \in \mathrm{T}$ and every production plan $\varepsilon \in \mathrm{E}$ must be an integer as close as possible to its ideal production $\lambda_{i,\varepsilon} t$.

Note also that the regular ideal production ($\lambda_{i,\varepsilon} t$) is defined here as the proportional quota that a product ($i \in I$) has over time, which is measured in cycles ($t = 1, .., |\mathrm{T}|$) and dedicated to manufacturing in a production plan ($\varepsilon \in \mathrm{E}$). Following the above and adopting the terminology that is

appropriate to the *Apportionment Problem* (ApP), we establish the following:

DEFINITION 1: Given the tuple $(I, \mathrm{T}, \mathrm{E}, \lambda_{i,\varepsilon})$, we will indicate that the sequence $\pi_\varepsilon(T) = (\pi_{1,\varepsilon}, \dots, \pi_{T,\varepsilon})$ is *Quota* if it satisfies all of the restrictions pmr (1), which is equivalent to saying that it has the property of preserving the production mix.

Sometimes the characterizations of regularity by restrictions (cr.1) is not sufficient to discriminate between a set of regular sequences in production because, for instances of industrial dimensions, the number of *Quota* sequences is usually very high. Therefore, it is convenient to adopt a characterization by objective function (cr.2) both to measure the regularity of the sequences and to offer more demanding discriminants. To accomplish this goal, we will refer to the functions used in the Just-in-Time ideology [11].

Specifically, to measure the non-regularity of a sequence $\pi_\varepsilon(T)$, we will use the sum of the quadratic discrepancies (Eq. 2) between the ideal production $(\lambda_{i,\varepsilon}t)$ and the real productions $(X_{i,t,\varepsilon})$ in every partial manufacturing sequence $(\pi_\varepsilon(t))$. As in [10], we adopt the metric $\Delta_{Q_X}(\varepsilon)$:

$$\Delta_{Q_X}(\varepsilon) = \sum_{t=1}^{T}\sum_{i=1}^{|I|}\left(X_{i,t,\varepsilon} - \lambda_{i,\varepsilon}t\right)^2 \quad \forall \varepsilon \in E \tag{2}$$

Other possible metrics for the non-regularity of the production of a sequence are those that correspond to the sum of absolute deviations (rectangular distances) or to the sum of Euclidean distances – see (3) and (4), respectively – between real and ideal productions.

$$\Delta_{R_X}(\varepsilon) \equiv \Delta_R(X, \varepsilon) = \sum_{t=1}^{T}\sum_{i=1}^{|I|}\left|X_{i,t,\varepsilon} - \lambda_{i,\varepsilon}t\right| \quad \forall \varepsilon \in E \tag{3}$$

$$\Delta_{E_X}(\varepsilon) \equiv \Delta_E(X, \varepsilon) = \sum_{t=1}^{T}\sum_{i=1}^{|I|}\sqrt{\left(X_{i,t,\varepsilon} - \lambda_{i,\varepsilon}t\right)^2} \quad \forall \varepsilon \in \mathrm{E} \tag{4}$$

Obviously, the minimization of the non-regularity $\Delta_{Q_X}(\varepsilon)$ in a manufacturing sequence can contradict the minimization of the work overload $W(T, \varepsilon)$. In fact, a problem of sequences that accounts for both objectives should be interpreted as a bi-objective problem.

3 MMSP-W/Δ_{Q_X} with preservation of the production mix and forced interruption of operations

We present a model for a variant of the MMSP-W sequencing problem that accounts for two types of aspects:

A.1 Economic: (i) objective function to minimize the work overload $(W(T))$ and (ii) incorporation of the idle time of the processors $(U(T))$.

A.2 Technical-productive: (i) objective function to minimize the non-regularity of the product manufacturing $(\Delta_{Q_X}(\varepsilon))$, (ii) productive operations subject to forced interruption to simplify the management of the production line and (iii) restrictions pmr of the production mix preservation to facilitate the management of Production Control.

We will call this new problem MMSP-W/Δ_{Q_X}/pmr/forced, and we will describe it as follows.

We are given the following:

– The set of product types $(I: i = 1, .., |I|)$ and the set of workstations $(K: k = 1, .., |K|)$;

- The cycle time c and the temporary windows l_k $(k \in K)$, which are granted to each processor to work on a product unit at its station, and the number of processors assigned to each station b_k $(k \in K)$; and
- The processing times $p_{i,k}$ $(i \in I \wedge k \in K)$ of the operations, which are measured at normal activity $(\alpha^N = 1)$, and the demand vectors $\vec{d} = (d_1, \dots, d_{|I|})$ and production mix $\vec{\lambda} = (\lambda_1, \dots, \lambda_{|I|})$, where d_i is the number of product units of type $i \in I$ contained in the production-demand plan, and λ_i is the proportion of the model $i \in I$ in the plan, which satisfy $\vec{\lambda} = \vec{d}/D$ and $T \equiv D = \sum_{\forall i} d_i$ -.

The problem is finding a *Quota* sequence of T products $\pi(T) = (\pi_1, \dots, \pi_T)$ with minimum work overload $W(\pi(T))$ and minimum non-regularity of manufacture $\Delta_Q(X, \pi(T))$ that satisfies the demand plan represented by the vector $\vec{d}$, while forcing the interruption of operations between the limits established by cycle c and the temporary windows l_k $(k \in K)$. The formulation of the model is as follows:

$$\min \mathcal{F}(\pi(T)) \equiv W(\pi(T)) \prec \Delta_{Q_X}(\pi(T)) \tag{5}$$

$$W(\pi(T)) = \sum_{t=1}^{T} \sum_{k=1}^{|K|} b_k w_{k,t}(\pi_t) \tag{6}$$

$$U(\pi(T)) = \sum_{t=1}^{T} \sum_{k=1}^{|K|} b_k u_{k,t}(\pi_t) \tag{7}$$

$$\Delta_{Q_X}(\pi(T)) = \sum_{t=1}^{T} \sum_{i=1}^{|I|} (X_{i,t} - \lambda_i t)^2 \tag{8}$$

$$w_{k,t}(\pi_t) = max(0, s_{k,t}(\pi_t) + p_{\pi_t,k} - (k + t - 2)c - l_k) \quad \forall k \in K \; \forall t = 1, .., T \tag{9}$$

$$u_{k,t}(\pi_t) = s_{k,t}(\pi_t) - e_{k,t-1}(\pi_{t-1}) \quad \forall k \in K \; \forall t = 1, .., T \tag{10}$$

$$X_{i,t} = |\{\pi_\tau \in \pi(t) = (\pi_1, \dots, \pi_t) \subseteq \pi(T) : \pi_\tau = i \in I\}| \quad \forall i \in I \; \forall t = 1, .., T \tag{11}$$

$$s_{k,t}(\pi_t) = max(e_{k,t-1}(\pi_{t-1}), e_{k-1,t}(\pi_t), (k + t - 2)c) \quad \forall k \in K \; \forall t = 1, .., T \tag{12}$$

$$e_{k,t-1}(\pi_{t-1}) = s_{k,t-1}(\pi_{t-1}) + p_{\pi_{t-1},k} - w_{k,t-1}(\pi_{t-1}) \quad \forall k \in K \; \forall t = 2, .., T \tag{13}$$

$$e_{k-1,t}(\pi_t) = s_{k-1,t}(\pi_t) + p_{\pi_t,k-1} - w_{k-1,t}(\pi_t) \quad \forall k \in K - \{1\} \; \forall t = 1, .., T \tag{14}$$

$$e_{k,t}(\pi_t) = min(s_{k,t}(\pi_t) + p_{\pi_t,k}, (k + t - 2)c + l_k) \quad \forall k \in K \; \forall t = 1, .., T \tag{15}$$

$$\lfloor \lambda_i t \rfloor \leq X_{i,t} \leq \lceil \lambda_i t \rceil \quad \forall i \in I \; \forall t = 1, .., T \tag{16}$$

$$X_{i,T} = d_i \quad \forall i \in I \tag{17}$$

In the model, the identity (5) expresses the minimization of the objective function $\mathcal{F}(\pi(T))$ that attends to two hierarchical criteria: the first $W(\pi(T))$ corresponds to the global work overload generated by the sequence, which is determined according to (6), and the second $\Delta_{Q_X}(\pi(T))$ is associated with the non-regularity of the production of the sequence, which is determined according to (8). The expression (7) is used to calculate the global idle time of the processors $U(\pi(T))$. The equalities (9) allow us to determine the partial work overloads of each station k and of each period t, with forced interruption of the operations, triggering it when the upper limit of the temporary window is reached: $(k + t - 2)c + l_k$. On the other hand, the equalities (10) define the partial idle time that is

generated in each station and in each manufacturing period as a function of $\pi(T)$, while (11) serves to count the number of products of type $i \in I$ in the partial sequence $\pi(t) \subseteq \pi(T)$. The equalities (12), on one side, and (13), (14) and (15), on the other, determine the minimum starting, $s_{k,t}$, and completion, $e_{k,t}$, instances of the $|K| \times D$ operations. The conditions (16) impose the *Quota* property on the manufacturing sequence $\pi(T)$. Finally, the equalities (17) impose the satisfaction of the demand plan ($d_i \ \forall i \in I$).

4 GRASP Algorithm

GRASP is a multi-start metaheuristic ([16], [17]) with two phases. The first, called the constructive phase, provides an initial solution through a randomized greedy procedure. The second, called the improvement phase, uses local search procedures to reach the local optima in one or more specific neighborhoods.

After a prefixed number of iterations (construction phase plus improvement phase), GRASP obtains the best manufacturing sequence $\pi(T) = (\pi_1, \dots, \pi_T)$, with a forced interruption of the operations, which satisfies the *Quota* property, and attending in this scheme, to the hierarchical bi-objective: minimum work overload and minimum non-regularity of production.

4.1 Phase 1: Construction of a sequence

Similar to [10], we construct a sequence of models $\pi(T) = (\pi_1, \dots, \pi_T)$, which assign progressively at each stage t $(t = 1, \dots, T)$ a product from the $CL(t)$ list of candidates that can be drawn to occupy the position t of the manufacturing sequence (t-th launch to line). Consequently, when stage t is reached, it is added to the partial sequence consolidated in the previous stage, $\pi(t-1) = (\pi_1, \dots, \pi_{t-1})$, a product $i \in CL(t)$ (see Algorithm 1).

For a product type $i \in I$ to enter the list $CL(t)$ of stage t, it must meet the following conditions: (i) the product does not have its demand fulfilled: $X_{i,t-1} < d_i$; (ii) it satisfies the *Quota* property in cycle t: $\left(\lfloor \lambda_i t \rfloor \leq X_{i,t-1} + 1 \leq \lceil \lambda_i t \rceil\right) \wedge \left(\left\lfloor \lambda_j t \right\rfloor \leq X_{j,t-1} \leq \left\lceil \lambda_j t \right\rceil, \forall j \neq i\right)$; and (iii) to the greatest extent possible, the product that enters the list cannot randomly displace other critical products, which results in the violation of the *Quota* property in later stages.

Condition (iii) is equivalent to imposing that the n-th unit of type $i \in I$ (n_i) must be launched to line in a cycle t_{n_i} contained in the interval $[t_{min}(n_i), t_{max}(n_i)]$, where $t_{min}(n_i)$ and $t_{max}(n_i)$ are the first and last cycle, respectively, in which the value n_i meets the *Quota* property.

If the list $CL(t)$ is empty by imposing the 3 conditions at the same time, then the list is opened for all products that do not have their demand fulfilled at stage t (condition (i)).

Once the $CL(t)$ list is built, we order the candidate products $i \in CL(t)$ according to two hierarchical priority indexes; these are P-1 and P-2.

P-1. Work overload: associated with the sequence $\pi_i(t) \equiv \pi(t-1) \cup \{i\}$, which results from adding the product $i \in CL(t)$ to the consolidated partial sequence $\pi(t-1)$:

$$f_i^{(t)} \equiv W(\pi_i(t)) = W(\pi(t-1)) + \sum_{k=1}^{m} b_k w_{k,t}(i) \quad \forall i \in CL(t) \wedge \forall t = 1, .., T \tag{18}$$

where $w_{k,t}(i)$ symbolizes the partial work overload supported by a processor of station $k \in K$ when the t-th product unit is of type i. If we consider the forced interruption of the operations, $w_{k,t}(i)$ it is calculated recursively while applying the formulas (19)-(22).

$$w_{k,t}(i) = max\left(0, s_{k,t}(i) + p_{i,k} - (k + t - 2)c - l_k\right) \tag{19}$$

$$s_{k,t}(i) = max\left(e_{k,t-1}(\pi_{t-1}), e_{k-1,t}(i), (k + t - 2)c\right) \tag{20}$$

$$e_{k,t-1}(\pi_{t-1}) = s_{k,t-1}(\pi_{t-1}) + p_{\pi_{t-1},k} - w_{k,t-1}(\pi_{t-1}) \tag{21}$$

$$e_{k-1,t}(i) = s_{k-1,t}(i) + p_{i,k-1} - w_{k-1,t}(i) \tag{22}$$

In (19), $s_{k,t}(i)$ is the start time of the operation in station k when a model of type i occupies the t-th position in the sequence. Note that $s_{k,t}(i)$ depends on the start of the t-th manufacturing cycle in station k ($(k+t-2)c$) and the instants in which the operations in progress are considered to be completed in station k ($e_{k,t-1}(\pi_{t-1})$) and in station $k-1$ ($e_{k-1,t}(i)$). To start these recursive calculations, we make $s_{1,1}(i) = 0\ \forall i \in I$.

P-2. Non-regularity of production: associated with the sequence $\pi_i(t) \equiv \pi(t-1) \cup \{i\}$, which results from adding the product $i \in CL(t)$ to the consolidated partial sequence $\pi(t-1)$:

$$g_i^{(t)} \equiv \Delta_{Q_X}(\pi_i(t)) = \Delta_{Q_X}(\pi(t-1)) + \sum_{j=1}^{|I|} \left(X_{j,t} - \lambda_j t\right)^2 \quad \forall i \in CL(t) \wedge \forall t = 1,..,T \tag{23}$$

where $\Delta_{Q_X}(\pi(t-1))$ is the non-regularity that is associated with the partial sequence of products $\pi(t-1)$ of stage $t-1$, and the values $X_{j,t}$ are the productions in process of the various models $j \in I$ up to stage t. Obviously, it is true that $X_{i,t} = X_{i,t-1} + 1$ and $X_{j,t} = X_{j,t-1}, \forall j \neq i$.

In contrast to [10], here, we have implemented, as a second priority index P-2, the contribution of a product to the function $\Delta_{Q_X}(\pi(T))$ of the non-regularity of the sequence $\pi(T)$; therefore, we have sacrificed the minimization of the idle time $U(\pi(T))$, which is calculated according to (5), as the second optimization criterion.

With the help of the indexes $W(\pi_i(t))$ and $\Delta_{Q_X}(\pi_i(t))$, we increasingly order the products from the list $CL(t)$ and convert it into the list $\overline{CL}(t)$. Considering that such ordering is applied hierarchically, the non-regularity $\Delta_{Q_X}(\pi_i(t))$ intervenes only when there is a tie in the partial work overload $W(\pi_i(t))$.

After this ordering, the list $\overline{CL}(t)$ is reduced through a mechanism that is a function of the admission factor Λ (percentage of products that can be drawn among the best candidates). After this operation, we obtain the restricted list $\overline{RCL}(t, \Lambda)$, which coincides with $\overline{CL}(t)$ when $\Lambda = 100\%$. The construction phase concludes with obtaining the sequence $\hat{\pi}(T)$ (see Algorithm 1).

Algorithm 1 An algorithm for the constructive phase of the sequence of products: $\hat{\pi}(T)$

1: // *Initialization*
2: **input** $\Lambda, I, K, D, c, (d_i, p_{i,k}, l_k)\ \forall i \in I\ \forall k \in K$
3: **initialize** $T = D, t = 0, \pi(t) = \{\emptyset\}, (n_i = 0, \lambda_i = d_i/D)\ \forall i \in I$
4: // *Create the candidate set*
5: **while** $(t \leq T)$ **do**
6: **set** $t = t + 1$
7: **set** $CL(t) = \{i \in I: (n_i < d_i) \wedge (t_{min}(n_i + 1) \leq t \leq t_{max}(n_i + 1))\}$
8: **if** $CL(t) = \{\emptyset\}$ **then**
9: **set** $CL(t) = \{i \in I: n_i < d_i\}$
10: **end if**
11: // *Evaluate alternative*
12: **for all** $(i \in CL(t))$ **do**
13: **set** $f_i^{(t)} \equiv W(\pi_i(t)) = W(\pi(t-1)) + \sum_{k=1}^{m} b_k w_{k,t}(i)$
14: **set** $g_i^{(t)} \equiv \Delta_{Q_X}(\pi_i(t)) = \Delta_{Q_X}(\pi(t-1)) + \sum_{j=1}^{|I|} \left(X_{j,t} - \lambda_j t\right)^2$
15: **end for**
16: // *Sort alternatives*
17: **sort** $CL(t)$: **set** $\overline{CL}(t)$ as the ordered list from $CL(t)$ according the $f_i^{(t)}$ and $g_i^{(t)}$ values
18: // *Select alternative*
19: **set** $pos_\Lambda = -int(-\Lambda \cdot |\overline{CL}(t)| \cdot RND)$

20: **set** $i_\Lambda = i \in \overline{CL}(t): pos_i = pos_\Lambda$
21: // *Update*
22: **set** $n_{i_\Lambda} \leftarrow n_{i_\Lambda} + 1$
23: **set** $\pi(t) = \pi(t-1) \cup \{i_\Lambda\}$
24: **end while**
25: // *Quota Property*
26: **set** $t = 0$
27: **while** $(t \leq T)$ **do**
28: **set** $t = t + 1$
29: **for all** $(i \in I)$ **do**
30: **set** $X_{i,t} = |\{\pi_\tau \in \pi(t) = (\pi_1, \dots, \pi_t) \subseteq \pi(T): \pi_\tau = i \in I\}|$
31: **if** $\lfloor \lambda_i t \rfloor \leq X_{i,t} \leq \lceil \lambda_i t \rceil$ **then**
32: **set** $quota = true$
33: **else**
34: **set** $quota = false$
35: **exit while**
36: **end if**
37: **end for**
38: **set** $\hat{\pi}(t) = \pi(t)$
39: **end while**
40: **if** $quota = false$ **then**
41: **solve** $MAXSAT$: **set** $\hat{\pi}(T) \longleftarrow \text{maxsat}(\pi(T), \lfloor \lambda_i t \rfloor \leq X_{i,t} \leq \lceil \lambda_i t \rceil)$
42: **end if**
43: // *End Algorithm* 1

Due to possible ties between products, according to their instants $t_{max}(n_i)$, during the construction phase of GRASP, the sequence of products $\pi(T)$ (see Lines 1-24 from Algorithm 1) can violate the *Quota* property in some production cycle. This circumstance occurs when the list $CL(t)$ is empty, and it must open the way to all products with pending demand. Under these conditions, we activate an exchange procedure that solves the problem of maximum satisfaction of restrictions $\left[t_{min}(n_i) \leq t_{n_i} \leq t_{max}(n_i), \forall n_i = 1,., d_i: i \in I\right]$, which provides as a solution a sequence $\hat{\pi}(T)$ that does satisfy the *Quota* property in all of the cycles.

The objective of MAXSAT procedure (see Line 41 from Algorithm 1) is getting a sequence $\hat{\pi}(T)$ (from $\pi(T)$), which meet all restrictions corresponding to the *Quota* property (16). The formulation of the model associated with MAXSAT procedure is as follows.

MAXSAT-*Quota_Sequence* Model:

$$\min Z = \sum_{i=1}^{|I|} \sum_{t=1}^{T} \left(z_{i,t}^{+} + z_{i,t}^{-}\right) \tag{24}$$

Subject to:

$$\sum_{i=1}^{|I|} x_{i,t} = 1 \quad \forall t = 1,., T \tag{25}$$

$$\sum_{t=1}^{T} x_{i,t} = d_i \quad \forall i \in I \tag{26}$$

$$X_{i,t} - \sum_{\tau=1}^{t} x_{i,\tau} = 0 \quad \forall i \in I \; \forall t = 1,., T \tag{27}$$

$$x_{i,t} = \begin{Bmatrix} 1 \Leftrightarrow \hat{\pi}_t = i \\ 0 \Leftrightarrow \hat{\pi}_t \neq i \end{Bmatrix} \quad \forall i \in I \; \forall t = 1,..,T \tag{28}$$

$$z_{i,t}^{+} = \begin{Bmatrix} 1 \Leftrightarrow X_{i,t} > \lceil \lambda_i t \rceil \\ 0 \Leftrightarrow X_{i,t} \leq \lceil \lambda_i t \rceil \end{Bmatrix} \quad \forall i \in I \; \forall t = 1,..,T \tag{29}$$

$$z_{i,t}^{-} = \begin{Bmatrix} 1 \Leftrightarrow X_{i,t} < \lfloor \lambda_i t \rfloor \\ 0 \Leftrightarrow X_{i,t} \geq \lfloor \lambda_i t \rfloor \end{Bmatrix} \quad \forall i \in I \; \forall t = 1,..,T \tag{30}$$

Where $z_{i,t}^{+}$ is a binary variable equal to 1 if $X_{i,t} > \lceil \lambda_i t \rceil$ $(i \in I, t = 1,..,T)$ and to 0 otherwise, and $z_{i,t}^{-}$ is a binary variable equal to 1 if $X_{i,t} < \lfloor \lambda_i t \rfloor$ $(i \in I, t = 1,..,T)$ and to 0 otherwise.

4.2 Phase 2: Improvement of the solution through local search

As in [10], we start with a *Quota* $\hat{\pi}(T)$ sequence and the local improvement phase, in which we proceed to consecutively and repetitively execute 4 descent algorithms in 4 neighborhoods until none of them improves the best solution that is achieved during the iteration. From two *Quota* sequences, we select the one that offers the least total work overload $W(\hat{\pi}(T))$ and, if the comparison is a tie, then the one with the least non-regularity of production $\Delta_{Q_X}(\hat{\pi}(T))$ is selected. The descent algorithms are based on the exchange and insertion of products, and they are oriented to the exploration of sequence cycles in both increasing and decreasing order. The descent algorithms are:

(i) *Forward exchange*: for all t position of the current sequence, $\hat{\pi}(T)$, it is determined the product type that is in that position and it is searched the next closest position $t'(t' > t)$ that is occupied by the same type $(\hat{\pi}_t = \hat{\pi}_{t'})$; if there is not this type at any position: $t' = T + 1$. After, the tentative exchange between $\hat{\pi}_t$ and the elements from the range $[t + 1, t' - 1]$ of the sequence is made. The first exchange that reduces the overall work overload, $W(\hat{\pi}(T))$, or the non-regularity of production, $\Delta_{Q_X}(\hat{\pi}(T))$, is consolidated whether the restrictions of production mix preservation are fulfilled for all products and positions from the range $[t + 1, t' - 1]$. While there is improvement this algorithm is repeated.

(ii) *Backward exchange:* for all t position of the $\hat{\pi}(T)$ sequence, the product type in the t position is detected and it is searched the previous closest position, $t'(t' < t)$, with the same product type, $\hat{\pi}_t = \hat{\pi}_{t'}$), as long as there is an improvement; if these type does not exist, it is considered $t' = 0$. Afterwards, the tentative exchange between $\hat{\pi}_t$ and the elements from the range $[t' + 1, t - 1]$ of the sequence is made. Thus, the first exchange is consolidated whereas the constraints of production mix preservation are met, and the overall work overload, $W(\hat{\pi}(T))$, or non-regularity of production, $\Delta_{Q_X}(\hat{\pi}(T))$, is reduced.

(iii) *Forward insertion:* for all t position of the current sequence, $\hat{\pi}(T)$, the product type in the t position is detected and it is searched the next closest position $t'(t' > t)$ that is occupied by the same type $(\hat{\pi}_t = \hat{\pi}_{t'})$; if these product type does not exist, it is considered $t' = T + 1$. Following, the $\hat{\pi}_t$ product is inserted in the range of sequence positions $[t + 1, t' - 1]$. Then, the first insertion that leads to reduce the work overload, $W(\hat{\pi}(T))$, or in case of tie, the non-regularity of production, $\Delta_{Q_X}(\hat{\pi}(T))$, is done, whenever the restrictions of production mix preservation are fulfilled for all product and for all position of the range $[t + 1, t' - 1]$. This action is repeated while there is improvement.

(iv) *Backward insertion*: for all t position of the current sequence, $\hat{\pi}(T)$, the product type in the t position is detected and it is searched the previous closest position $t'(t' < t)$ that is occupied by the same type $(\hat{\pi}_t = \hat{\pi}_{t'})$; if these product type does not exist, it is considered $t' = 0$. Following, the $\hat{\pi}_t$ product is inserted in the range of sequence positions $[t' + 1, t - 1]$. Then, it is carried out the first insertion that leads to reduce the work overload, $W(\hat{\pi}(T))$, or (*ex aequo*) the non-regularity of production, $\Delta_{Q_X}(\hat{\pi}(T))$. All consolidated insertion must meet the restrictions of production mix preservation for all product and for all position of $[t' + 1, t - 1]$ range. This action is repeated while there is improvement.

5 Case study

5.1 Data

The computational experience that we propose is focused on analyzing the behavior of the GRASP algorithm proposed in this paper, compared to two other procedures, in terms of the quality of the solutions and the CPU times; these procedures are (1) BDP-1 [6] and (2) GRASP-1 [7]. As in [8], [9] and [10], the analysis is conducted on a case study of the Nissan plant in Barcelona: an assembly line of 9 types of engines grouped into 3 families (SUVs - Sport Utility Vehicle-, vans and trucks). Figure 1 shows an engine type M1 that belongs to the Sport Utility Vehicle family.

Fig. 1. Nissan Pathfinder Engine. Characteristics: (i) 747 parts and 330 references, (ii) 378 elemental assembly tasks grouped in 140 production line tasks.

In this production line, 42 operators work in shifts of 8 hours, subject to a cycle time of 175 seconds. Briefly, the data that frames the case taken here are as follows:

- Number of workstations: $|K| \equiv m = 21$;
- Number of product types: $|I| = 9$ $(i = 1,..,9)$;
- Cycle time: $c = 175\ s.$, and a temporary window: $l_k = 195\ s.$ $(\forall k = 1,..,21)$;
- Number of homogeneous processors (with 2 operators): $b_k = 1$ $(\forall k = 1,..,21)$;
- Processing time $p_{i,k}$, according to the engine type and workstation $(\forall i \in I, \forall k \in K)$, with values between $89\ s.$ and $185\ s.$ at normal activity (see Table 3 in Annex I);
- Number of engine demand plans: $|E| = 23$ $(\varepsilon = 1,..,23)$, corresponding to the Nissan-9Eng.I instances. All of the plans have the same daily demand (see Table 4 in Annex I); and
- Daily demand: $T \equiv D_\varepsilon = 270\ units$ $(\forall \varepsilon = 1,..,23)$.

5.2 Procedures and algorithms

With reference to resolution procedures, we have considered the following alternatives: the Mixed Integer Linear Programming (MILP), the Greedy and Randomized Adaptive Search Procedure (GRASP), the Bounded Dynamic Programming (BDP), the hybrid procedure that combines GRASP with the Linear Programming (GRASP-LP), and the hybrid procedure that combines the BDP with the Linear Programming (BDP-LP).

These procedures present a set of strengths and weaknesses in accordance with the following five qualities: (i) guarantee of achieving optimal solutions, (ii) memory requirement, (iii) ease of implementation, (iv) quality of solutions, and (v) the CPU time. Specifically:

- GRASP offers high quality solutions in reduced CPU times. It is an easily implementable procedure and requires low memory but it does not guarantee optimal solutions. It is efficient in the forced variant, while it requires the assistance of LP to solve the free variant (GRASP-LP).
- BDP offers high quality solutions with CPU times similar to those used by the MILP. Its implementation is highly labor and requires more memory than GRASP by less than MILP. In addition, it can guarantee optimum results and offer lower bounds. Like GRASP, the BDP is efficient for solving the MMSP_Γ(forced) variants but it requires the LP to solve the MMSP_Γ(free) variants (BDP-LP).
- MILP offers high quality solutions in reasonable CPU times (two hours). It is the procedure easier to implement, but it requires more memory than GRASP and BDP. It can guarantee optimal solutions and give lower bounds for instances. It solves efficiently the free variant, while it is inefficient for solving the forced one, so we have avoided using MILP in this work.

The case study we takes has been solved by three algorithms: BDP-1, GRASP-1 and GRASP-2. Then we describe briefly these three algorithms.

BDP-1 [6] is an algorithm based on the Bounded Dynamic Programming procedure (BDP).For his hand, BDP is a procedure based on Dynamic Programming which employs lower and upper Bounds to reduce the space search of solutions.

Similar to [18], we can build a linked graph without loops or direct cycles of $T+1$ stages. The set of vertices in level t ($t = 0,.,T$) will be noted as $J(t)$. All vertex level t, $J(t,j)(j = 1,.,|J(t)|$, is defined by the triad $(\vec{q}(t,j), \vec{e}(t,j), W(t,j))$, where:

- $\vec{q}(t,j) \equiv (q_1(t,j),.,q_{|I|}(t,j)$ represents the vector of demand satisfied.
- $\vec{e}(t,j) \equiv (e_1(t,j),.,e_{|K|}(t,j)$ represents the vector of finishing instants of the operations at the workstations.
- $W(t,j)$ represents the total non-completed work (or work overload) over the operations.

At level 0 of the graph, there is only one $J(0)$ vertex. Initially, we may consider that at level t, $J(t)$ contains the vertices associated with all of sub-sequences that can be built with t products. However, it is easy to reduce the cardinal that $J(t)$ may present a priori, establishing the following relationship of dominance and equivalence:

$$J(t,j) \prec J(t,j') \Leftrightarrow [\vec{q}(t,j) = \vec{q}(t,j)] \wedge [\vec{e}^c(t,j) \prec \vec{e}^c(t,j')] \wedge [W(t,j) \leq W(t,j')] \quad (31)$$

$$J(t,j) \equiv J(t,j') \Leftrightarrow [\vec{q}(t,j) = \vec{q}(t,j)] \wedge [\vec{e}^c(t,j) = \vec{e}^c(t,j')] \wedge [W(t,j) = W(t,j')] \quad (32)$$

Where $\vec{e}^c(t,j)$ is the vector of corrected finishing instants in accordance with the cycle time. That is: $e_k^c(t,j) = max\{e_k(t,j), (t+k-1)c\}\ \forall t = 1,.,D\ \wedge\ \forall k \in K$.

Under these conditions, finding a sequence that minimize the work overload $W(\pi(T))$ (see (6)) is equivalent to finding an optimum path from vertex $J(0)$ to the set of vertices $J(T)$ of stage T through the reduced graph by dominance (31) and equivalence (32). Therefore, any algorithm of extreme paths in the graphs is valid for finding solutions to the proposed problem.

However, realistic industrial problems where I and T (9 engine types and 270 engines) are large give rise to graphs with a large number of vertices. Therefore, we recommend resorting to procedures that do not explicitly require the presence of all of the vertices for calculation.

BDP-1 algorithm consists of generating a part of the graph from level 0 to level T, one level at a time. The generated vertices may potentially form a part of an optimum path (from 0 to T) that is based on the construction of an optimum segment of t stages, from $J(0)$ to $J(t,j)$, and on the evaluation of a lower bound of the complement $R(t,j)$ to reach stage T, for example $LB1(t,j)$ from [6].

BDP-1 only keeps the information of two consecutive stages in memory, t and $t+1$ ($t = 0,.,T-1$), for which it uses the following lists of vertices $VL(t)$ and $VL(t+1)$, respectively:

- List $VL(t)$ contains information about the vertices consolidated in stage t that can potentially form part of an optimum or good quality path.
- List $VL(t+1)$ contains the vertices that are tentatively generated one-by-one from each vertex of list $VL(t)$ through the possible transitions between stages t and $t+1$.

Although the use of $VL(t)$ and $VL(t+1)$ notably reduces memory needs, the number of vertices that can be generated for a stage can be very large. Therefore, we impose a limitation on the number of $H(t)$ vertices stored in stage t. This limitation, called window width, is represented as H, $H(t) \leq H$ $(t = 1,.,T)$. In addition, we set the maximum number of transitions from a vertex to the value $|I|$, which is, in addition, the maximum value possible.

Evidently, some vertices tentatively generated in stage t will not be recorded in list $VL(t+1)$. In effect, we use the following rules:

1. ***Remove*** the $J(t+1,j)$ vertex when the value of its lower bound, $LBZ = W(t+1,j) + LB1(t+1,j)$, is greater than or equal to the value of a known solution Z_0 (upper bound).
2. ***Reject*** the $J(t+1,j)$ when already exist other vertex, in $VL(t+1)$, which is dominant or equivalent.
3. ***Discard*** the placement of $J(t+1,j)$ vertex on the list $VL(t+1)$ when the list is full and $J(t+1,j)$ has a lower bound LBZ that is greater than or equal to the largest of the lower bounds of the vertices already recorded in $VL(t+1)$, although an optimum path may pass through $J(t+1,j)$.
4. ***Replace*** the $J(t+1,h)$ vertex from $VL(t+1)$ by the $J(t+1,j)$ vertex when it dominates the first, or when $J(t+1,j)$ has a lower bound less than the corresponding to $J(t+1,h)$ and $H(t+1) = H$, although the optimum path may pass through the moved vertex.

To obtain an initial solution with value Z_0 (upper bound of the value of the optimum solution), it is sufficient to use a Greedy procedure or BDP with a small window width (v.gr.- $H = 1$).

When the procedure ends $(t = T)$, we can initially find two possible situations:

- List $VL(T)$ is empty, which means that we are unable to find a solution with a value less than Z_0.
- List $VL(T)$ is not empty, which means that $VL(T)$ contains at least one vertex, $J(T,h)$, whose work overload, $W(t,h)$, is less than Z_0.

Also, if list $VL(T)$ is not empty, we can guarantee that any vertex $J(T,j) \in VL(T)$ has an associated sequence, $\pi(J(T,j))$, that is optimal in any of the following cases: (i) $max_{0\leq t\leq T} H(t) < H$; or (ii) $[max_{0\leq t\leq T} H(t) = H] \wedge [W(T,h) \leq LBZ_{min}]$, where LBZ_{min} is the best lower bound of the set of vertices replaced during the execution of algorithm.

For its part, GRASP-1 [7] is the first GRASP algorithm that appears in the literature to solve the problem MMSP-*W*/*U*/pmr/forced (Mixed model sequencing problem with Work overload and Useless time minimization, production mix restrictions and forced interruption of operations).

In the Construction Phase of GRASP-1, the list of candidates is prioritized by means of two hierarchical functions: the Work overload $W(\pi_i(t))$ (formula (18)) and the Useless time $U(\pi_i(t))$. The $U(\pi_i(t))$ values are calculated according to (33) and (34).

$$U(\pi_i(t)) = U(\pi(t-1)) + \sum_{k=1}^{m} b_k u_{k,t}(i) \quad \forall i \in CL(t) \wedge \forall t = 1,..,T \tag{33}$$

where $u_{k,t}(i)$ is the useless time that a processor of k workstation has between the finalisation instant, $e_{k,t-1}(\pi_{t-1})$, of the unit in the $t-1$ position of the sequence and the start instant, $s_{k,t}(i)$, of the unit, which is of i type. That is:

$$u_{k,t}(i) = s_{k,t}(i) - e_{k,t-1}(\pi_{t-1}) \tag{34}$$

GRASP-1 considers the minimization of the Work overload $W(\pi_i(t))$ as the first objective and the minimization of Useless time $U(\pi_i(t))$ as second objective. GRASP-1 does not take into account, at any time, the objective of minimizing the function of Non-regularity of production $\Delta_{Q_X}(\pi(T))$

(formula (8)), although the incorporation to GRASP-1 of the production mix restrictions (formula (16)) favors the reduction of $\Delta_{Q_X}(\pi(T))$.

For its part, GRASP-2 (this work) is the first GRASP algorithm to solve the problem MMSP-W/Δ_{Q_X}/pmr/forced (Mixed model sequencing problem with Work overload and Non-regularity of production minimization, production mix restrictions and forced interruption of operations).

GRASP-2 considers the minimization of the Work overload $W(\pi(T))$ as the first objective and the minimization of quadratic function of Non-regularity of production $\Delta_{Q_X}(\pi(T))$ as second objective. GRASP-2 does not take into account, at any time, the objective of minimizing the Useless time $U(\pi(T))$ (formula (7)). Similarly to GRASP-1, the incorporation to GRASP-2 of the production mix restrictions (formula (16)) favors the reduction of $\Delta_{Q_X}(\pi(T))$.

5.3 Codes

The compiled codes of the procedures that we have selected in this work have the following characteristics:

- BDP-1: Algorithm under the scheme of Bounded Dynamic Programming (BDP) focusing on minimizing the work overload $W(\pi(T))$ (see [6]). The code has been compiled and executed on an iMac (Intel Core 2 Duo 2.33 GHz, 3 GB RAM). The maximum number of transitions from each vertex is equal to the number of product types $|I| = 9$. The window widths for each demand plan are $H = (1,10,50,100,250,500,750,1000)$, which generates 184 executions of the algorithm. In each execution, the initial solution Z_0 for H_n is equal to the best solution obtained with H_{n-1}, except for $H_1 = 1$, where $Z_0 \to \infty$. The average CPU time (accumulated for 8 values of H) used for each demand plan equals 7664.27 *s*. The *Quota* property has not been imposed on the sequences, and the interruption of operations is forced.
- GRASP-1: The GRASP algorithm focused on hierarchically minimizing the work overload $W(\pi(T))$ and the idle time $U(\pi(T))$ (see [7]). The code has been compiled and executed on an iMac (Intel Core i7 2.93 GHz, 8 GB RAM). The maximum number of iterations for each plan is equal to 10 with three candidate admission factors $\Lambda = (25\%, 50\%, 100\%)$, which generates 690 solutions in 69 executions. The average CPU time used per demand plan equals 425.3 s. The *Quota* property has been imposed on all sequences, and the interruption of operations is forced.
- GRASP-2: the GRASP algorithm focused on hierarchically minimizing the work overload $W(\pi(T))$ and the non-regularity of production $\Delta_{Q_X}(\pi(T))$ (this paper). The code has been compiled and executed on an iMac (Intel Core i7 2.93 GHz, 8 GB RAM). The maximum number of iterations for each demand plan is equal to 20 with three candidate admission factors $\Lambda = (25\%, 50\%, 100\%)$, which generates 1380 solutions in 69 executions. The average CPU time used per iteration is equal to 82.8 s., and the average time required to obtain the best solution in each demand plan is equal to 679.5 s. The *Quota* property has been imposed on all sequences, and the interruption of operations is also forced.

5.4 Results

Table 1 shows the best results achieved by BDP-1 (see Table 7 in [6]), GRASP-1 (see table 2 in [7]) and GRASP-2 (this paper), for the work overload W in the 23 demand plans $\varepsilon \in$ E.

Table 1 also shows the winning algorithm in each demand plan and the unitary gains of GRASP-2 versus BDP-1 (ΔG2vB), GRASP-2 versus GRASP-1 (ΔG2vG1) and BDP-1 versus GRASP-1 (ΔBvG1), which are determined according to (35).

$$\Delta\mathcal{P}v\mathcal{P}'(\varepsilon) = \frac{W_{\mathcal{P}'}(\varepsilon) - W_{\mathcal{P}}(\varepsilon)}{min\left(W_{\mathcal{P}'}(\varepsilon), W_{\mathcal{P}}(\varepsilon)\right)} \quad \forall \varepsilon \in E, \forall \mathcal{P} \in \{\mathrm{G2}, \mathrm{B}\}, \forall \mathcal{P}' \in \{\mathrm{B}, \mathrm{G1}\} \tag{35}$$

Considering the work overload and from the analysis of Table 1, we can affirm the following:

- In terms of the number of solutions, BDP-1 and GRASP-2 tie with the 11 best solutions out of 23, while GRASP-1 achieves the best solution 4 times. GRASP-1 and GRASP-2 tie in plan 10, and the three procedures obtain the same result on the work overload for plan 23.
- GRASP-2 beats BDP-1 in 12 plans, loses in 10 and ties in one. The average unitary gain of BDP-1 over GRASP-2 is 29.7% when BDP-1 is the winner, while that of GRASP-2 over BDP-1 is 14% when GRASP-2 wins. On a global average, the unitary gain of BDP-1 over GRASP-2 is 5.6%.
- GRASP-2 beats GRASP-1 in 17 plans, loses in 3 and ties in 3. The global average unitary gain of GRASP-2 over GRASP-1 is on the order of 2.3%. In detail, GRASP-2 beats GRASP-1 with a partial average unitary gain of 3.5%, and GRASP-1 partially beats GRASP-2 with a gain on the order of 2.8%.
- BDP-1 beats GRASP-1 10 times out of 23, loses in 12 plans and ties in one. The partial average unitary gains, when BDP-1 beats GRASP-1 and vice versa, are equal to 35.7% and 12.3%. BDP-1 globally beats GRASP-1 with a gain of 9.1%.
- BDP-1, GRASP-1 and GRASP-2 needed on average 7664.3 s, 425.3 s and 679.5 s, respectively, to confirm their best solution in each demand plan.

$\varepsilon \in$ E	W_{BDP1}	W_{GRASP1}	W_{GRASP2}	ΔG2vB	ΔG2vG1	ΔBvG1	W_{BG12}	$Winning$
1	166	142	152	0.09	-0.07	-0.17	142	G1
2	464	404	392	0.18	0.03	-0.15	392	G2
3	432	436	436	-0.01	0.00	0.01	432	BDP
4	440	535	515	-0.17	0.04	0.22	440	BDP
5	897	868	871	0.03	-0.00	-0.03	868	G1
6	663	748	739	-0.11	0.01	0.13	663	BDP
7	823	790	784	0.05	0.01	-0.04	784	G2
8	129	96	86	0.50	0.12	-0.34	86	G2
9	1149	1235	1234	-0.07	0.00	0.07	1149	BDP
10	1249	1246	1246	0.00	0.00	-0.00	1246	G1/G2
11	50	124	116	-1.32	0.07	1.48	50	BDP
12	369	284	280	0.32	0.01	-0.30	280	G2
13	379	399	403	-0.06	-0.01	0.05	379	BDP
14	578	543	535	0.08	0.01	-0.06	535	G2
15	553	461	452	0.22	0.02	-0.20	452	G2
16	223	255	227	-0.02	0.12	0.14	223	BDP
17	640	556	548	0.17	0.01	-0.15	548	G2
18	962	1067	1063	-0.10	0.00	0.11	962	BDP
19	980	971	969	0.01	0.00	-0.01	969	G2
20	104	234	208	-1.00	0.13	1.25	104	BDP
21	854	943	939	-0.10	0.00	0.10	854	BDP
22	1104	1084	1078	0.02	0.01	-0.02	1078	G2
23	107	107	107	0.00	0.00	0.00	107	ALL
Average	-	-	-	-0.056	0.023	0.091	-	-

Table 1: For each plan $\varepsilon \in$ E, Work overload $W(\varepsilon)$ according to procedure $(W_{BDP1}, W_{GRASP1}, W_{GRASP2})$. Unitary gain between pairs of procedures $(\Delta\text{G2vB}, \Delta\text{G2vG1}, \Delta\text{BvG1})$, best solution W_{BG12} and Winning Algorithm.

On the other hand, in Table 2, we collect the Hamilton's lower bounds $(\Delta_{Q_X}(\varepsilon)_H)$ of the non-regularity function $\Delta_{Q_X}(\varepsilon)$ as well as the values offered by GRASP-1 [7] and GRASP-2 for $\Delta_{Q_X}(\varepsilon)$; we also show the Winning algorithm, the unitary gains of GRASP-2 versus GRASP-1 (ΔG2vG1) and the relative distances of these to the lower bounds (ΔHvG1 and ΔHvG2) that are determined according to (36).

$$\Delta\mathcal{P}v\mathcal{P}'(\varepsilon) = \frac{\Delta_{Q_X}(\varepsilon)_{\mathcal{P}'} - \Delta_{Q_X}(\varepsilon)_{\mathcal{P}}}{min(\Delta_{Q_X}(\varepsilon)_{\mathcal{P}'}, \Delta_{Q_X}(\varepsilon)_{\mathcal{P}})} \quad \forall \varepsilon \in E, \forall \mathcal{P} \in \{\text{H}, \text{G2}\}, \forall \mathcal{P}' \in \{\text{G1}, \text{G2}\} \tag{36}$$

We must note that BDP-1 does not incorporate any mechanism to reduce the non-regularity; therefore, it would be unfair to compare it with the GRASP procedures in this regard; moreover, we want to highlight that BDP-1 solves a problem that is less restricted than GRASP-1 and GRASP-2, and

therefore, it is better suited to obtain better values for the work overload W.

Focusing on the results achieved by GRASP-1 and GRASP-2 for $\Delta_{Q_X}(\varepsilon)$, we have the following:

- GRASP-2 beats GRASP-1 in 18 plans, loses in 4 and ties in one with an optimal value (plan #1). The average unitary gain of GRASP-2 over GRASP-1 is 5.4% when GRASP-2 is the winning procedure, while that of GRASP-1 over GRASP-2 is only 2% when the former one wins. On a global average, the unitary gain of GRASP-2 over GRASP-1 is 3.8%.
- On average, GRASP-1 offers results that are at 54.4% of the Hamilton's averaged limit $\Delta_{Q_X}(\varepsilon)_H$, while GRASP-2 is at 48.7% of that averaged limit. This finding assumes an advantage in the regularity of production of GRASP-2 over GRASP-1 of 5.7%.
- On average, GRASP-1 is 1.6 times faster than GRASP-2.

$\varepsilon \in$ E	$\Delta_{Q_X}(\varepsilon)_H$	$\Delta_{Q_X}(\varepsilon)_{G1}$	$\Delta_{Q_X}(\varepsilon)_{G2}$	ΔHvG1	ΔHvG2	ΔG2vG1	$\Delta_{Q_X}(\varepsilon)_{G12}$	$Winning$
1	400.00	400.00	400.00	0.00	0.00	0.00	400.00	G1/G2
2	250.70	397.01	376.81	0.58	0.50	0.05	376.81	G2
3	263.70	380.60	368.15	0.44	0.40	0.03	368.15	G2
4	263.06	396.43	404.54	0.51	0.54	-0.02	396.43	G1
5	247.92	429.02	389.42	0.73	0.57	0.10	389.42	G2
6	262.78	395.67	379.71	0.51	0.44	0.04	379.71	G2
7	269.17	403.35	394.28	0.50	0.46	0.02	394.28	G2
8	248.31	414.02	384.14	0.67	0.55	0.08	384.14	G2
9	252.31	394.76	399.30	0.56	0.58	-0.01	394.76	G1
10	249.35	415.47	358.11	0.67	0.44	0.16	358.11	G2
11	247.97	429.02	419.86	0.73	0.69	0.02	419.86	G2
12	239.93	416.24	384.93	0.73	0.60	0.08	384.93	G2
13	239.82	419.79	406.54	0.75	0.70	0.03	406.54	G2
14	237.86	408.90	385.10	0.72	0.62	0.06	385.10	G2
15	259.96	401.11	380.51	0.54	0.46	0.05	380.51	G2
16	248.94	388.06	380.59	0.56	0.53	0.02	380.59	G2
17	249.91	391.60	382.74	0.57	0.53	0.02	382.74	G2
18	245.07	402.56	404.42	0.64	0.65	-0.00	402.56	G1
19	266.48	373.52	356.22	0.40	0.34	0.05	356.22	G2
20	268.80	386.50	350.91	0.44	0.31	0.10	350.91	G2
21	312.50	409.83	407.50	0.31	0.30	0.01	407.50	G2
22	249.76	382.68	400.48	0.53	0.60	-0.05	382.68	G1
23	267.96	377.08	368.56	0.41	0.38	0.02	368.56	G2
Average	-	-	-	0.54	0.49	0.04	-	-

Table 2: For each plan $\varepsilon \in$ E, Non-regularity $\Delta_{Q_X}(\varepsilon)$ according to procedure $(\mathrm{H}, \mathrm{G1}, \mathrm{G2})$. Unitary gain between pairs of procedures $(\Delta\mathrm{HvG1}, \Delta\mathrm{HvG2}, \Delta\mathrm{G2vG1})$, best solution $\Delta_{Q_X}(\varepsilon)_{G12}$ and Winning Algorithm.

To summarize the above results (tables 1 and 2) we present by two box plots (Figures 2 and 3) the variation of Work overload $W(\varepsilon)$ (6), and the variation of the unitary gain functions $(\Delta\mathrm{HvG1}, \Delta\mathrm{HvG2}, \Delta\mathrm{G2vG1})$ for the $\Delta_{Q_X}(\varepsilon)$ function of non-regularity of production (36).

From the Box-plots we can state the following:

- Concerning the Work overload, all values $(\mathrm{W_{BDP1}}, \mathrm{W_{GRASP1,}}\ \mathrm{W_{GRASP2}})$ are between the normal boundaries established by 1.5 times the interquartile range or the minimum and maximum values of each dataset. Therefore the unity gains (columns $\mathrm{W_{BDP1}}, \mathrm{W_{GRASP1,}}\ \mathrm{W_{GRASP2}}$ from Table 1) do not present outliers.
- According the $\Delta_{Q_X}\big(\pi(T)\big)$ metric and assessing GRASP-2 against GRASP-1 (column ΔG2vG1 in Table 1), we can see a very narrow the interquartile range and a clearly biased value distribution; also, the box plot shows (Figure 2) outliers at all the procedure comparisons.
- The $\Delta_{Q_X}\big(\pi(T)\big)$ metric when the GRASP-2 procedure is assessed against GRASP-1 has made evident the superiority of GRASP-2.

– Finally, the comparison between Halmilton's lower bound (H) against the other two procedures presents more advantage on GRASP-1 than on GRASP-2.

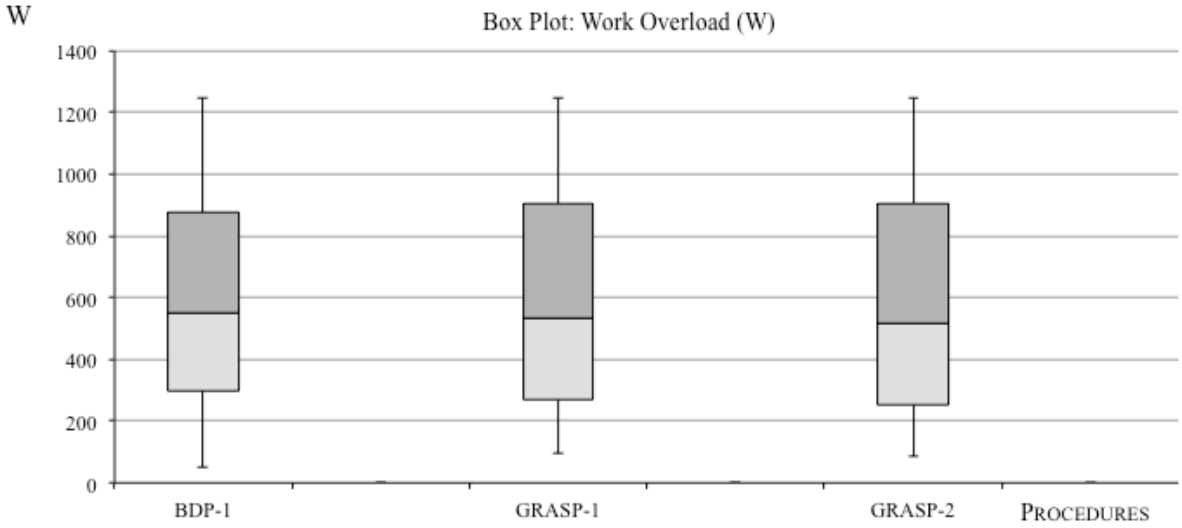

Fig. 2. Box-plot for the best results achieved by BDP-1, GRASP-1 and GRASP-2, for the work overload $W(\varepsilon)$ in the set of instances Nissan-9Eng.I (23 plans $\varepsilon \in$ E)

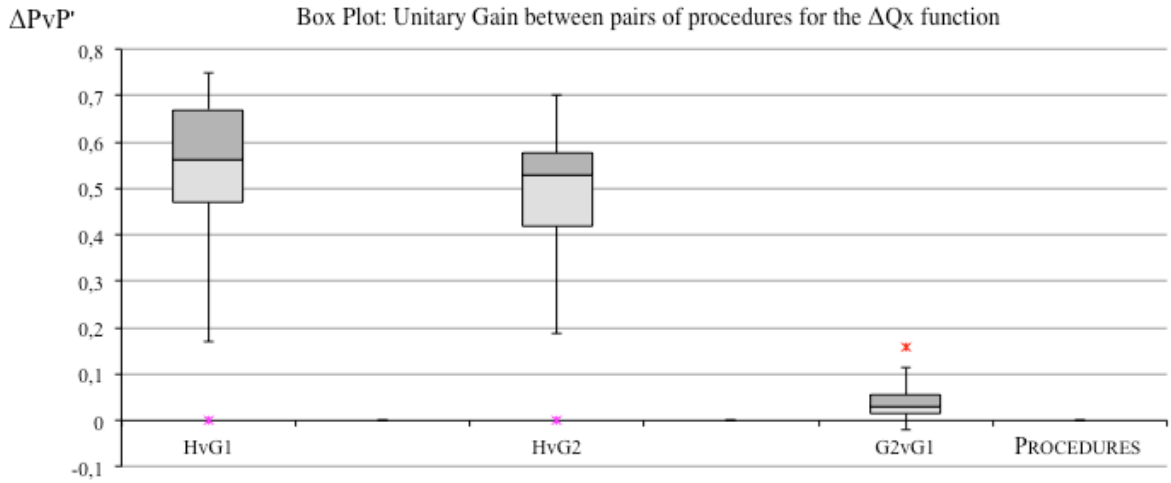

Fig. 3. Box-plot for Unitary Gains between two different procedures (ΔHvG1, ΔHvG2, ΔG2vG1) for the $\Delta_{Qx}(\varepsilon)$ function of non-regularity of production in the set of instances Nissan-9Eng.I (23 plans $\varepsilon \in$ E)

To compare the GRASP algorithms, next we use a test statistics tests to assess whether the observed differences between compared algorithms are statistically significant. This analysis is based on the relative distance (or Gap) between solution offered by GRASP-1 and GRASP-2 against solutions given by BDP-1.

$$Gap(\mathcal{P},\varepsilon) = \frac{W_{\mathcal{P}}(\varepsilon) - W_{BDP-1}(\varepsilon)}{W_{\mathcal{P}}(\varepsilon)} \quad \forall \varepsilon \in E, \forall \mathcal{P} \in \{\mathrm{G1}, \mathrm{G2}\} \tag{37}$$

Through the Kolmogorov-Smirnov normality test we have asses normality of both data, $Gap(G1,\varepsilon)$ and $Gap(G2,\varepsilon)$. With a p-value equal to 0.047 for $Gap(G1,\varepsilon)$ and 0.036 for $Gap(G2,\varepsilon)$, we can reject the null hypothesis and conclude that the data are non-normal with $\alpha = 0.05$.

Accordingly, we perform a nonparametric test for paired samples by calculating the differences and then performing a one-sample Wilcoxon signed rank test. The Wilcoxon signed rank test for

paired data gives us a p-value equal to 0.005. Therefore, we can reject the null hypothesis and we can state that the mean ranks differ.

To determine whether the medians of results for the non-regularity given by GRASP-1 and GRASP-2 differ, we use the Mann-Whitney Test. This non-parametric test calculates allows us to reject the null hypothesis ($p-value = 0.013 < \alpha = 0.05$), H0: The median of the non-regularity given by GRASP-1 equals the median of the non-regularity given by GRASP-2. Specifically, we can state that the difference between the medians is statistically significant.

Consequently, we can affirm that GRASP-1 and GRASP-2 are different algorithms and that we cannot rule out either of them. Despite the advantage of GRASP-2 over GRASP-1 in average value, both in work overload $W(\pi(T))$ and in $\Delta_{Q_X}(\pi(T))$, GRASP-1 obtains better solutions than GRASP-2 in some occasions.

We can also conclude that the advantage of GRASP-2 over GRASP-1 is due to the initial sequence obtained in Phase 1 (construction of the sequence). Therefore, the tiebreaker treatment in work overload through the non-regularity function is more effective than that corresponding to the useless time function.

6 Conclusions

Despite the disadvantage that the GRASP-1 and GRASP-2 procedures have against BDP-1, because the latter two solve a more restricted problem with the purpose of obtaining *Quota* sequences, the implemented GRASP algorithms appear to be competitive in terms of the work overload W, versus BDP-1.

The particular competition between GRASP-2 and GRASP-1 is resolved in favor of the former, both terms of both the work overload (with an average unitary gain of 2.3%) and the improvement of the regularity of production (5.7% averaged difference with the Hamilton limit); nevertheless, the latter procedure is 1.6 times faster than the former, which is in part because of the maximum number of iterations granted to each procedure in each demand plan.

Keeping in mind that this work corresponds to a first stage of investigation, in which the parameters for both GRASP-1 and GRASP-2 have been calibrated based on the results achieved by these procedures in 5 iterations per demand plan, some other aspects are required to balance the competition between GRASP-1 and GRASP-2.

First, further experiments should test imposing an identical maximum number of iterations per demand plan in both procedures. Second, the values of the candidate admission factors (Λ) in the construction phase of the sequences, which are currently $\Lambda = (25\%, 50\%, 100\%)$, should be expanded, for both GRASP-1 and GRASP-2.

Other possible improvements in the GRASP algorithms, which are left to be explored, are in the descent algorithms that affect the improvement phase of the solution through local search. In fact, currently in both procedures, the product exchanges and insertions are oriented to the exploration, in increasing and decreasing order, of the production cycles (t), with the possibility of orienting such exploration to the product types $i \in I$.

Another future experiment is to compare BDP-1 with versions of GRASP-1 and GRASP-2 that do not impose the *Quota* property on the manufacturing sequences. This comparison would allow a fair competition, from a computational point of view, between the three procedures because the three are focused on minimizing a single objective: the work overload W.

Finally, it is still pending to hybridize GRASP-2 with other procedures that allow obtaining solutions for the problem MMSP-*W*/pmr/free (see [10]) from the proposed problem in this paper, which we call MMSP-W/Δ_{Q_X}/pmr/forced. For this purpose, we think that an assistant procedure to GRASP could be linear programming, as shown in [10], or alternatively, the design of heuristic algorithms that, from a manufacturing sequence perspective, reduce its work overload by intelligent disruption of the operations.

Acknowledgments. This work was funded by the Ministry of Economy and Competitiveness (Government of de Spain) through project TIN2014-57497-P (FHI-SELM2).

References

[1] Yano, C.A., Rachamadugu, R.: Sequencing to Minimize Work Overload in Assembly Lines with Product Options, Management Science 37(5), 572-586 (1991). DOI: 10.1287/mnsc.37.5.572

[2] Bolat, A., Yano, C.A.: Scheduling algorithms to minimize utility work at a single station on a paced assembly line. Production Planning & Control 3(4), 393–405 (1992). DOI: 10.1080/09537289208919409

[3] Tsai, L.H.: Mixed-Model Sequencing to Minimize Utility Work and the Risk of Conveyor Stoppage. Management Science 41(3), 485–495 (1995). DOI: 10.1287/mnsc.41.3.485

[4] Boysen, N., Fliedner, M., Scholl, A.: Sequencing mixed-model assembly lines: Survey, classification and model critique. European Journal of Operational Research 192(2), 349–373 (2009). DOI: 10.1016/j.ejor.2007.09.013

[5] Scholl, A., Klein, R., Domschke, W.: Pattern Based Vocabulary Building for Effectively Sequencing Mixed-Model Assembly Lines. Journal of Heuristics 4(4), 359-381 (1998). DOI: 10.1023/A:1009613925523

[6] Bautista, J., Cano, A.: Solving mixed model sequencing problem in assembly lines with serial workstations with work overload minimisation and interruption rules. European Journal of Operational Research 210(3), 495-513 (2011). DOI: 10.1016/j.ejor.2010.10.022

[7] Bautista, J., Alfaro-Pozo, R., Batalla-García, C.: GRASP for sequencing mixed models in an assembly line with work overload, useless time and production regularity. Progress in Artificial Intelligence 5(1), 27-33 (2016). DOI 10.1007/s13748-015-0071-z

[8] Bautista, J., Cano, A., Alfaro, R.: Modeling and solving a variant of the mixed-model sequencing problem with work overload minimisation and regularity constraints. An application in Nissan's Barcelona Plant. Expert Systems with Applications, 39(12), 11001–11010 (2012). DOI: 10.1016/j.eswa.2012.03.024

[9] Bautista, J., Cano, A., Alfaro-Pozo, R.: A hybrid dynamic programming for solving a mixed-model sequencing problem with production mix restriction and free interruptions. Progress in Artificial Intelligence 6(1), 27–39 (2017). DOI: 10.1007/s13748-016-0101-5

[10] Bautista, J., Alfaro-Pozo, R.: Free and regular mixed-model sequences by a linear program-assisted hybrid algorithm GRASP-LP. Progress in Artificial Intelligence 6(2), 159-169 (2017). DOI:10.1007/s13748-017-0110-z

[11] Monden, Y.: Toyota Production System: An Integrated Approach to Just-In-Time. 4th Edition. New York, Productivity Press (2011).

[12] Aigbedo, H., Monden, Y.: A parametric procedure for multicriterion sequence scheduling for just-in-time mixed-model assembly lines. International Journal of Production Research 35, 2543–2564 (1997). DOI: 10.1080/002075497194651

[13] Fullerton, R.R., Kennedy, F.A., Widener, S.K.: Lean manufacturing and firm performance: The incremental contribution of lean management accounting practices. Journal of Operations Management 32(7-8), 414-428 (2014). DOI: 10.1016/j.jom.2014.09.002

[14] Miltenburg, J.: Level Schedules for Mixed-Model Assembly Lines in Just-In-Time Production Systems. Management Science, 35(2), 192-207 (1989). DOI: 10.1287/mnsc.35.2.192

[15] Bautista J., Cano A., Alfaro R., Batalla C.: Impact of the Production Mix Preservation on the ORV Problem. In: Bielza C. et al. (eds) Advances in Artificial Intelligence. CAEPIA 2013. Lecture Notes in Computer Science, vol 8109. Springer, Berlin, Heidelberg (2013). DOI: 10.1007/978-3-642-40643-0_26

[16] Feo, T.A., Resende, M.G.C.: Greedy Randomized Adaptive Search Procedures. Journal of Global Optimization 6(2), 109-133 (1995). DOI: 10.1007/BF01096763

[17] Resende M.G., Ribeiro C.C.: Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications. In: Gendreau M., Potvin JY. (eds) Handbook of Metaheuristics. International Series in Operations Research & Management Science, vol 146. Springer, Boston, MA (2010). DOI: 10.1007/978-1-4419-1665-5_10

[18] Bautista, J., Companys, R., Corominas, A.: Heuristics and exact algorithms for solving the Monden problem. Eur. J. Oper. Res. **88**(1), 495–513 (1996). DOI: 10.1016/0377-2217(94)00165-0

Appendix I: Data of the set of instances Nissan-9Eng.I

$k\backslash i$	M1	M2	M3	M4	M5	M6	M7	M8	M9
1	104	100	97	92	100	94	103	109	101
2	103	103	105	107	101	108	106	102	110
3	165	156	164	161	148	156	154	164	155
4	166	175	172	167	168	167	168	156	173
5	111	114	114	115	117	117	115	111	111
6	126	121	122	124	127	130	120	121	134
7	97	96	96	93	96	89	94	101	92
8	100	97	95	106	94	102	103	102	100
9	179	174	173	178	178	171	177	171	174
10	178	172	172	177	178	177	175	173	175
11	161	152	168	167	167	166	172	157	177
12	96	106	105	97	101	100	96	104	96
13	99	101	102	101	99	101	96	102	99
14	147	155	142	154	146	143	154	153	155
15	163	152	156	152	153	152	154	156	156
16	163	185	183	178	169	173	172	182	171
17	173	179	178	169	173	178	174	175	175
18	176	167	181	180	172	173	173	168	184
19	162	150	152	152	160	151	155	148	167
20	164	161	157	159	162	160	162	158	157
21	177	161	154	168	172	170	167	149	169

Table 3. Processing times at normal activity $(p_{i,k})$ in seconds for the 9 engine types $(i \in I)$ in the 21 workstations $(k \in K)$ of the set of instances Nissan-9Eng.I.

$\varepsilon \in$ E	M1	M2	M3	M4	M5	M6	M7	M8	M9	SUV	Van	Truck	Total
1	30	30	30	30	30	30	30	30	30	90	60	120	270
2	30	30	30	45	45	23	23	22	22	90	90	90	270
3	10	10	10	60	60	30	30	30	30	30	120	120	270
4	40	40	40	15	15	30	30	30	30	120	30	120	270
5	40	40	40	60	60	8	8	7	7	120	120	30	270
6	50	50	50	30	30	15	15	15	15	150	60	60	270
7	20	20	20	75	75	15	15	15	15	60	150	60	270
8	20	20	20	30	30	38	38	37	37	60	60	150	270
9	70	70	70	15	15	8	8	7	7	210	30	30	270
10	10	10	10	105	105	8	8	7	7	30	210	30	270
11	10	10	10	15	15	53	53	52	52	30	30	210	270
12	24	23	23	45	45	28	28	27	27	70	90	110	270
13	37	37	36	35	35	23	23	22	22	110	70	90	270
14	37	37	36	45	45	18	18	17	17	110	90	70	270
15	24	23	23	55	55	23	23	22	22	70	110	90	270
16	30	30	30	35	35	28	28	27	27	90	70	110	270
17	30	30	30	55	55	18	18	17	17	90	110	70	270
18	60	60	60	30	30	8	8	7	7	180	60	30	270
19	10	10	10	90	90	15	15	15	15	30	180	60	270
20	20	20	20	15	15	45	45	45	45	60	30	180	270
21	60	60	60	15	15	15	15	15	15	180	30	60	270
22	20	20	20	90	90	8	8	7	7	60	180	30	270
23	10	10	10	30	30	45	45	45	45	30	60	180	270

Table 4. Daily demands by product and plan $(d_{i,\varepsilon})$ for the 23 instances Nissan-9Eng.I ($\varepsilon \in$ E).